***1)***

a) Thanh ghi PC dùng để lưu địa chỉ của lệnh sẽ được nạp vào

b) Instruction Memory chứa các lệnh của chương trình

* Input: địa chỉ lệnh từ PC
* Output: lệnh

c) Registers là tập hợp của 32 thanh ghi 32 bit, mỗi thanh ghi được xác định bởi số hiệu 5 bit

* Input: đầu vào để chọn thanh ghi cần đọc (Read register 1, Read register 2), cần ghi (Write register), đưa dữ liệu ghi vào thanh ghi (Write data ), tín hiệu điều khiển ghi dữ liệu vào thanh ghi (Regwrite)
* Output: đầu ra đọc dữ liệu từ thanh ghi 32bit (Read data 1, Read data 2).

d)

* Input: dữ liệu từ 2 thanh ghi nguồn được đọc 2 đầu ra hoặc dữ liệu của lệnh từ instruction memory khi đi qua bộ mở rộng dấu.
* Output: kết quả phép toán

e)

* Bộ Control:
* Main Control: nhận input là 6 bit opcode từ 32 bit lệnh (trường Instruction [31-26]).
* ALU Control: nhận input là 6 bit opcode từ 32 bit lệnh, 6 bit fuction.
* Output:
* Main Control: 10 tín hiệu điều khiển cho datapath
* ALU Control: ALUCtrl cho bộ ALU

f) Data Memory dùng để chứa dữ liệu dùng cho lệnh load và store

* Input: dữ liệu ghi vào (Write data), lựa chọn ô nhớ để truy cập và sử dụng dữ liệu (Address), cho phép đọc ghi (MemRead, MemWrite), xung clock đồng bộ thao tác ghi
* Output: dữ liệu xuất ra khi chọn MemRead

g) Mux dùng để lựa chọn đầu vào cho 1 thành phần nào đó khi ta có nhiều hơn hoặc bằng 2 lựa chọn đầu vào.

* *Ví dụ:* ta thêm 1 mux để lựa chọn toán hạng thứ 2 cho ALU từ Read data 2 hoặc từ bộ mở rộng hằng số 16 ->32 bit. Khi đó nếu như giá trị mux=0 thì đầu vào thứ 2 của ALU sẽ là Read data 2, ngược lại mux=1 thì đầu vào là từ bộ mở rộng hằng số

h) Sign-extend (mở rộng có dấu) dùng để mở rộng hằng số imm16 bit thành hằng số 32 bit

* *Ví dụ:* giá trị 16 bit: 11111111 11001000 (-56)

=> sign-extended 32 bit: 11111111 11111111 11111111 11001000

giá trị 16 bit: 00000000 01001000 (72)

=> sign-extended 32 bit: 00000000 00000000 00000000 01001000

***2)***

* RegDst: chọn lựa thanh ghi từ Rt hoặc Rd để đưa vào RW của Registers
* RegWrite: cho phép ghi kết quả ALU result vào thanh ghi
* MemRead: yêu cầu đọc (lw) từ Data memory
* MemWrite: yêu cầu ghi(sw) từ Data memory
* MemtoReg: chọn dữ liệu của BusW là kết quả từ ALU result hoặc từ
* Memory Data\_out
* Branch: lựa chọn rẽ nhánh
* Jump: lựa chọn nhảy
* ALUSrc: chọn toán hạn thứ 2 của ALU từ Read data 2 hoặc Extender

***3)***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | RegDst | Jump | Branch | MemRead | MemtoReg | MemWrite | ALUSrc | RegWrite |
| lw $s0, 8($a0) | 0 | x | x | 1 | 1 | 0 | 1 | 1 |
| sw $s0, 8($a0) | x | x | x | 0 | x | 1 | 1 | 0 |
| add $s0, $s1, $s2 | 1 | x | x | 0 | 0 | 0 | 0 | 1 |
| beq $t2, $t1, label | x | 0 | 1 | 0 | x | 0 | 0 | 0 |
| j label | x | 1 | 0 | 0 | x | 0 | x | 0 |

***4)***

a)

• Load: Instruction memory -> MUX -> Registers -> MUX -> ALU -> Data memory -> MUX -> Registers

=>Thời gian hoàn thành: 200 + 10 + 150 + 10 + 100 + 200 + 10 + 150 = 830 ns

(ta bỏ qua Sign-extend time vì MUX+ Register > Sign-extend)

• Store: Instruciont memory -> Registers -> MUX -> ALU -> Data memory

=>Thời gian hoàn thành: 200 + 150 + 10 + 100 + 200 = 660 ns

(ta bỏ qua Sign-extend time vì Register > Sign-extend)

• ALU: Instruction memory -> MUX -> Registers -> MUX -> ALU -> MUX -> Registers

=>Thời gian hoàn thành: 200 + 10 + 150 + 10 + 100+ 10 +150 = 630 ns

• Branch: Instruciont memory -> Registers -> MUX -> ALU -> MUX -> MUX

=>Thời gian hoàn thành: 200 + 150 + 10 + 100 + 10 + 10 = 480 ns

(ta bỏ qua Shift left time vì đó không phải đường đi dài nhất)

• Jump: Instruction memory -> Shift left -> Add -> MUX -> MUX

=>Thời gian hoàn thành: 200 + 10 + 10 + 10 + 10 = 240 ns

b) Thời gian cycle trong single clock cycle bằng thời gian thực hiện lệnh dài nhất => bằng thời gian thực hiện lệnh Load = 830 ns